Mathematics for Deep Learning

Partial Derivatives for Backpropagation

Inputs and Output

Let x denote an input vector. For the moment we'll leave its dimension and how we index its components unspecified.

Similarly, let y denote the output vector corresponding to the input vector, x. Its dimension and how we index its components will for a moment also remain unspecified, but note that the dimensions of x and y are generally completely different.

Neural Nets and Layers

We will conceive of a neural net as consisting of L layers. Each layer takes an input vector and produces an output vector. Then

$$y = x^{(L)} \leftarrow x^{(L-1)} \leftarrow x^{(L-2)} \leftarrow \ldots \leftarrow x^{(2)} \leftarrow x^{(1)} \leftarrow x^{(0)} = x$$

The parenthesized superscripts denote layer numbers, not exponents. A net with one layer ($L = 1$) has two vectors, consisting of one input vector, one output vector, and no vectors in between. An $L = 4$ neural net has five vectors $x^{(0)}$ to $x^{(4)}$.

We still haven't said what the dimensions of the vectors are or how we index their components, but again note that the dimensions of the various $x^{(l)}$, $l = 0, \ldots, L$ are in general unrelated.

Functions to Represent Layer Operations

The lth function representing the layer l operation transforms $x^{(l)}$ to $x^{(l+1)}$. Sacrificing compactness for clarity, we'll write this operation as $x^{(l+1)} = f^{(l+1 \leftarrow l)}\left(x^{(l)}\right)$.

In a neural net with L layers, there are L (not $L + 1$) functions f. The one that operates on the input vector is $f^{(1 \leftarrow 0)}\left(x^{(l)}\right)$, and the one producing the output vector is $f^{(L \leftarrow L-1)}\left(x^{(L-1)}\right)$.

Parameters

The function $f^{(l+1\leftarrow l)}$ also depends on parameters which we will call $\alpha^{(l)}$. So for completeness, we need to capture this too:

$$x^{(l+1)} = f^{(l+1\leftarrow l)}\left(x^{(l)}; \alpha^{(l)}\right)$$

When we chain the transformation of the L layers in the neural net together, the complete transformation is

$$x^{(L)} = f^{(L\leftarrow L-1)}\left(f^{(L-1\leftarrow L-2)}\left(\ldots f^{(2\leftarrow 1)}\left(f^{(1\leftarrow 0)}\left(x^{(0)}; \alpha^{(0)}\right); \alpha^{(1)}\right) \ldots; \alpha^{(L-2)}\right); \alpha^{(L-1)}\right)$$

The ellipses lack clarity, so a better way to see the pattern is to write out a non-trivial case, such as $L = 4$:

$$L = 4: \quad x^{(4)} = f^{(4\leftarrow 3)}\left(f^{(3\leftarrow 2)}\left(f^{(2\leftarrow 1)}\left(f^{(1\leftarrow 0)}\left(x^{(0)}; \alpha^{(0)}\right); \alpha^{(1)}\right); \alpha^{(2)}\right); \alpha^{(3)}\right)$$

This can be made even clearer by using $y \equiv x^{(4)}$ and $x \equiv x^{(0)}$:

$$y \equiv x^{(4)} = f^{(4\leftarrow 3)}\left(f^{(3\leftarrow 2)}\left(f^{(2\leftarrow 1)}\left(f^{(1\leftarrow 0)}\left(x; \alpha^{(0)}\right); \alpha^{(1)}\right); \alpha^{(2)}\right); \alpha^{(3)}\right)$$

Actually, let's write out the $L = 3$, $L = 2$, and $L = 1$ cases completely too, since that is barely more than a bit of copy-and-paste:

$$L = 3: \quad y \equiv x^{(3)} = f^{(3\leftarrow 2)}\left(f^{(2\leftarrow 1)}\left(f^{(1\leftarrow 0)}\left(x; \alpha^{(0)}\right); \alpha^{(1)}\right); \alpha^{(2)}\right)$$

$$L = 2: \quad y \equiv x^{(2)} = f^{(2\leftarrow 1)}\left(f^{(1\leftarrow 0)}\left(x; \alpha^{(0)}\right); \alpha^{(1)}\right)$$

$$L = 1: \quad y \equiv x^{(1)} = f^{(1\leftarrow 0)}\left(x; \alpha^{(0)}\right)$$

This very special form of composite function is central to the simplicity of the mathematics that follows.

Training

The neural net makes predictions that depend on $x = x^{(0)}$ by computing all the intermediaries $x^{(l)}$. However, during training, we are more keenly interested on the net's dependence on the parameters, $\alpha^{(l)}$, which are tuned on the training data to improve the predictions, than we are on the values of the $x^{(l)}$. A measure of goodness (or badness, which is usually called "loss") is computed, and then derivatives of the loss are taken w.r.t. (with respect to) the parameters, $\alpha^{(l)}$. Taking the derivatives of the loss function immediately requires us (after one application of the chain rule) to know the derivatives w.r.t. any of the parameters $\alpha^{(l)}$ of:

$$x^{(L)} = f^{(L\leftarrow L-1)}\left(f^{(L-1\leftarrow L-2)}\left(\ldots f^{(2\leftarrow 1)}\left(f^{(1\leftarrow 0)}\left(x^{(0)};\alpha^{(0)}\right);\alpha^{(1)}\right)\ldots;\alpha^{(L-2)}\right);\alpha^{(L-1)}\right)$$

Remember that each of the vectors $x^{(l)}$ has a still-unspecified dimension and each of the vectors $\alpha^{(l)}$ also has a still-unspecified dimension, and that none of these dimensions in general have anything to do with any of the others.

A Special Case

$L = 4$, and Entirely One-Dimensional

To get the hang of this, let's take a very special case. We'll do $L = 4$, and we'll assume that each and every one of $y = x^{(4)}$, $x^{(3)}$, $x^{(2)}$, $x^{(1)}$, and $x = x^{(0)}$, are one-dimensional, and that $\alpha^{(3)}$, $\alpha^{(2)}$, $\alpha^{(1)}$, and $\alpha^{(0)}$ also happen to each be one-dimensional. We want to know the derivatives w.r.t. any of $\alpha^{(3)}$, $\alpha^{(2)}$, $\alpha^{(1)}$, and $\alpha^{(0)}$ of

$$y \equiv x^{(4)} = f^{(4\leftarrow 3)}\left(f^{(3\leftarrow 2)}\left(f^{(2\leftarrow 1)}\left(f^{(1\leftarrow 0)}\left(x^{(0)};\alpha^{(0)}\right);\alpha^{(1)}\right);\alpha^{(2)}\right);\alpha^{(3)}\right)$$

Well, let's do one:

$$\frac{\partial y}{\partial \alpha^{(3)}} = \frac{\partial f^{(4\leftarrow 3)}}{\partial \alpha^{(3)}}$$

That wasn't so bad. Let's do another, but this one will require one application of the chain rule:

$$\frac{\partial y}{\partial \alpha^{(2)}} = \frac{\partial f^{(4\leftarrow 3)}}{\partial x^{(3)}}\,\frac{\partial f^{(3\leftarrow 2)}}{\partial \alpha^{(2)}}$$

Hopefully, you already see the pattern, but if not, write a few more out, and you will get:

$$\frac{\partial y}{\partial \alpha^{(1)}} = \frac{\partial f^{(4\leftarrow 3)}}{\partial x^{(3)}}\,\frac{\partial f^{(3\leftarrow 2)}}{\partial x^{(2)}}\,\frac{\partial f^{(2\leftarrow 1)}}{\partial \alpha^{(1)}}$$

$$\frac{\partial y}{\partial \alpha^{(0)}} = \frac{\partial f^{(4\leftarrow 3)}}{\partial x^{(3)}}\,\frac{\partial f^{(3\leftarrow 2)}}{\partial x^{(2)}}\,\frac{\partial f^{(2\leftarrow 1)}}{\partial x^{(1)}}\,\frac{\partial f^{(1\leftarrow 0)}}{\partial \alpha^{(0)}}$$

Re-Introducing Dimensionality

Sticking with $L = 4$, let's allow the vectors $y \equiv x^{(4)}$, $x^{(3)}$, $x^{(2)}$, $x^{(1)}$, and $x \equiv x^{(0)}$, to regain dimensionality, and similarly, the vectors $\alpha^{(3)}$, $\alpha^{(2)}$, $\alpha^{(1)}$, and $\alpha^{(0)}$ will also regain dimensionality.

We'll call the dimensions of the $x^{(l)}$ in the $L = 4$ case, d_4, d_3, d_3, and d_1 (in the partial derivatives, you will see that we don't need to refer to d_0), and the dimensions of the $\alpha^{(l)}$ in the $L = 4$ case, n_3, n_2, n_1, and n_o.

The i_lth component of $x^{(l)}$ will be $x^{(l)}_{i_l}$ and the μ_lth component of $\alpha^{(l)}$ will be $\alpha^{(l)}_{\mu_l}$. We want to know:

$$\frac{\partial y_{i_4}}{\partial \alpha^{(l)}_{\mu_l}} \equiv \frac{\partial x^{(4)}_{i_4}}{\partial \alpha^{(l)}_{\mu_l}}$$

Let's do $l = 3$ first because it is the easiest:

$$\frac{\partial y_{i_4}}{\partial \alpha^{(3)}_{\mu_3}} \equiv \frac{\partial x^{(4)}_{i_4}}{\partial \alpha^{(3)}_{\mu_3}} = \frac{\partial f^{(4\leftarrow 3)}_{i_4}}{\partial \alpha^{(3)}_{\mu_3}}$$

We will finally need one result from multi-variable calculus to actually write out the rest. Here is $l = 2$:

$$\frac{\partial y_{i_4}}{\partial \alpha^{(2)}_{\mu_2}} \equiv \frac{\partial x^{(4)}_{i_4}}{\partial \alpha^{(2)}_{\mu_2}} = \sum_{i_3} \frac{\partial f^{(4\leftarrow 3)}_{i_4}}{\partial x^{(2)}_{i_3}} \frac{\partial f^{(3\leftarrow 2)}_{i_3}}{\partial \alpha^{(2)}_{\mu_2}}$$

The summation is over i_3 which goes from 1 to d_3 (or if you prefer, 0 to $d_3 - 1$).

The indices i_4 and μ_2 in this expression are known as free indices, and the index i_3, which appears in exactly two places, is summed over its d_3 possible values.

From here on, whenever an index appears in two places, we will assume that it is summed over its possible values, and not bother with writing the summation symbol. Thus our result is simply

$$\frac{\partial y_{i_4}}{\partial \alpha^{(2)}_{\mu_2}} = \frac{\partial f^{(4\leftarrow 3)}_{i_4}}{\partial x^{(3)}_{i_3}} \frac{\partial f^{(3\leftarrow 2)}_{i_3}}{\partial \alpha^{(2)}_{\mu_2}}$$

Hopefully, you already see the pattern, but if not, write a few more out, and you will get:

$$\frac{\partial y_{i_4}}{\partial \alpha^{(1)}_{\mu_1}} = \frac{\partial f^{(4\leftarrow 3)}_{i_4}}{\partial x^{(3)}_{i_3}} \frac{\partial f^{(3\leftarrow 2)}_{i_3}}{\partial x^{(2)}_{i_2}} \frac{\partial f^{(2\leftarrow 1)}_{i_2}}{\partial \alpha^{(1)}_{\mu_1}}$$

$$\frac{\partial y_{i_4}}{\partial \alpha^{(2)}_{\mu_0}} = \frac{\partial f^{(4\leftarrow 3)}_{i_4}}{\partial x^{(3)}_{i_3}} \frac{\partial f^{(3\leftarrow 2)}_{i_3}}{\partial x^{(2)}_{i_2}} \frac{\partial f^{(2\leftarrow 1)}_{i_2}}{\partial x^{(1)}_{i_1}} \frac{\partial f^{(1\leftarrow 0)}_{i_1}}{\partial \alpha^{(2)}_{\mu_0}}$$

Keep in mind that there are 2 and 3 suppressed summations in these expressions, respectively.

Notice that for $l = 3$ we had zero x derivatives and one α derivative. For $l = 2$, we had one x derivative and one α derivative. For $l = 1$, we had two x derivatives and still one α derivative. Finally for $l = 0$, we had three x derivatives and still one α derivative.

The $L = 4$ pattern is that you have $4 - l - 1$ suppressed summations, the same number of x derivatives, and finally one α derivative.

Tensors (Actually just Matrices)

Tensors can have 0, 1, 2, or more indices and the number of indices is called the rank. When they have 0 indices, we call them scalars. When they have 1 index, we call them vectors. When they have 2 indices, we call them matrices. When they have r indices with $r > 2$, we call them tensors of rank r, but these are conventions. All these are tensors, just with different ranks, and the ones we have immediate use for are the rank 2 tensors, aka, matrices.

With matrices at our disposal, the expressions that are already getting messy in even the $L = 4$ case become matrix multiplications. We will define

$$M^{(l\leftarrow l-1)}_{i_l,i_{l-1}} \equiv \frac{\partial f^{(l\leftarrow l-1)}_{i_l}}{\partial x^{(l-1)}_{i_{l-1}}}$$

This definition hides the plethora of partial derivatives with respect to the x's. For example,

$$\frac{\partial y_{i_4}}{\partial \alpha^{(1)}_{\mu_1}} = \frac{\partial f^{(4\leftarrow 3)}_{i_4}}{\partial x^{(3)}_{i_3}} \frac{\partial f^{(3\leftarrow 2)}_{i_3}}{\partial x^{(2)}_{i_2}} \frac{\partial f^{(2\leftarrow 1)}_{i_2}}{\partial \alpha^{(1)}_{\mu_1}}$$

becomes

$$\frac{\partial y_{i_4}}{\partial \alpha^{(1)}_{\mu_1}} = M^{(4\leftarrow 3)}_{i_4,i_3} M^{(3\leftarrow 2)}_{i_3,i_2} \frac{\partial f^{(2\leftarrow 1)}_{i_2}}{\partial \alpha^{(1)}_{\mu_1}}$$

Let's keep going and also hide the partial derivatives with respect to the α's by defining

$$O^{(4\leftarrow 1)}_{i_4,\mu_1} \equiv \frac{\partial f^{(4\leftarrow 3)}_{i_4}}{\partial \alpha^{(1)}_{\mu_1}}$$

and

$$O^{(2\leftarrow 1)}_{i_2,\mu_1} \equiv \frac{\partial f^{(2\leftarrow 1)}_{i_2}}{\partial \alpha^{(1)}_{\mu_1}}$$

Putting those two definitions in, we have

$$O^{(4\leftarrow 1)}_{i_4,\mu_1} = M^{(4\leftarrow 3)}_{i_4,i_3} M^{(3\leftarrow 2)}_{i_3,i_2} O^{(2\leftarrow 1)}_{i_2,\mu_1}$$

Because **people that do matrix multiplications all the time...** know that (1) each matrix carries two indices, (2) that the left-most index in an expression is a free index (in this case, i_4), (3) that the right-most index in an expression is also free (in this case, μ_1), and (4a) that all the in-between indices are repeated twice, (4b) are dummy indices (in this case, i_3 and i_2), and (4c) are summed over all their

allowed values. Knowing this pattern, they **...don't bother writing any indices at all!** With those conventions, we have simply

$$O^{(4\leftarrow 1)} = M^{(4\leftarrow 3)}\, M^{(3\leftarrow 2)}\, O^{(2\leftarrow 1)}$$

Summary of Our Final Result in the General Case

The partial derivatives we need to know when doing gradient descent are given by matrix multiplications. If there are L layers and we are looking for the derivative of one of the components of y, specifically, y_{i_L}, w.r.t. one of the components of $\alpha^{(l)}$, specifically, $\alpha^{(l)}_{\mu_l}$, then we need to compute the following matrix product

$$O^{(L\leftarrow l)} = M^{(L\leftarrow L-1)}\, M^{(L-1\leftarrow L-2)} \ldots M^{(l+2\leftarrow l+1)}\, O^{(l+1\leftarrow l)}$$

where

$$M^{(l\leftarrow l-1)}_{i_l,i_{l-1}} \equiv \frac{\partial f^{(l\leftarrow l-1)}_{i_l}}{\partial x^{(l-1)}_{i_{l-1}}}$$

$$O^{(L\leftarrow l)}_{i_L,\mu_l} \equiv \frac{\partial y_{i_L}}{\partial \alpha^{(l)}_{\mu_l}}$$

and

$$O^{(l+1\leftarrow l)}_{i_{l+1},\mu_l} \equiv \frac{\partial f^{(l+1\leftarrow l)}_{i_{l+1}}}{\partial \alpha^{(l)}_{\mu_l}}$$

There are $L - l - 1$ of the M matrices in this product. Having taken all the products, which including the right-most O, is a total of $L - l$ products, we take the i_L, μ_lth entry.

This is insanely tidy for something that could have turned into a mess. The keys to the elegant and compact result were:

(1) the rather special assumed functional form of our neural net,

$$y \equiv x^{(L)} = f^{(L\leftarrow L-1)}\Big(f^{(L-1\leftarrow L-2)}\Big(\ldots f^{(2\leftarrow 1)}\Big(f^{(1\leftarrow 0)}\Big(x^{(0)};\, \alpha^{(0)}\Big);\, \alpha^{(1)}\Big)\ldots;\, \alpha^{(L-2)}\Big);\, \alpha^{(L-1)}\Big),$$

(2) our hiding of decently-complicated partial derivatives in matrices, and

(3) the conventions for matrix algebra, especially summation over repeated indices (which originates from Einstein's 1916 general relativity paper and is known as "the Einstein summation convention").

Example — Neural Nets for XOR or Fizz Buzz

I really ought to add a super-simple example to illustrate the general case, such as the neural net that Grus used to solve XOR or Fizz Buzz in his live coding session, https://youtu.be/o64FV-ez6Gw.

Appendix — Motivating the Ordinary and Multi-Variable Chain Rule

I could add some motivation for the one multi-variable calculus result that was required for the derivation, in which case the notes would only require ordinary calculus as a prerequisite. In fact, I could even motivate the ordinary chain rule, the proof of which requires a good understanding of limits.

For now, I will refer you first to 3Blue1Brown (Grant Sanderson) who has a nice explanation of the ordinary chain rule: https://youtu.be/YG15m2VwSjA. After having digested that, when Sanderson pictorially explains the same material that I am explaining in this writeup, he motivates the multi-variable chain rule: https://youtu.be/tIeHLnjs5U8.

It is important to be able to fluently go back and forth between precise notation and pictorial understandings if you want more than a hand-waving understanding of how backpropagation works, and if you want to be able to code backpropagation yourself without relying Keras, TensorFlow, or PyTorch.

Let me do a bit of multi-variable chain rule motivation:

Essentially, the multivariable chain rule says that if you go north 100' and are forced up 200' in elevation and then west 100' and forced up another 300' in elevation, then you could alternatively have gone northwest 141.4' and you would necessarily have been forced up 200'+300'=500' in elevation. There are perverse functions for which such perfectly intuitive addition becomes troublesome, but let's ignore those.

To be somewhat more general, the multivariable chain rule says that motions along perpendicular directions (called independent variables) simply add in their final effect. This addition is captured in sums over however many independent variables there are in the multivariable chain rule. This is why we had the sum over i_3 when we wrote

$$\frac{\partial x_{i_4}^{(4)}}{\partial \alpha_{\mu_2}^{(2)}} = \sum_{i_3} \frac{\partial f_{i_4}^{(4\leftarrow 3)}}{\partial x_{i_3}^{(2)}} \frac{\partial f_{i_3}^{(3\leftarrow 2)}}{\partial \alpha_{\mu_2}^{(2)}}$$

Multivariable calculus is awesome, and the high points are Gauss' Theorem and Stokes' Theorem, but you don't need either theorem for backpropagation. You only need the multi-variable chain rule.